Implementing Network Stack IPv4 and IPv6

The Network Stack IPv4 and IPv6 features are used to configure and monitor Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6).

This module describes the new and revised tasks you need to implement Network Stack IPv4 and IPv6 on your Cisco IOS XR network.

Note   For a complete description of the Network Stack IPv4 and IPv6 commands, refer to the *Network Stack IPv4 and IPv6 Commands* module of the *IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*. To locate documentation for other commands that appear in this chapter, use the *Cisco ASR 9000 Series Aggregation Services Router Commands Master List*, or search online.

Feature History for Implementing Network Stack IPv4 and IPv6

Release	Modification
Release 3.7.2	This feature was introduced.
Release 3.9.0	GRE for IPv4 feature was added.
Release 4.2.1	IPv6 over IPv4 GRE Tunnel interface feature was added.
Release 5.3.1	IPv6 Virtual Addresses feature was added.
Release 5.3.2	IPv6 Path MTU Discovery support for applications using Ping protocol was introduced.
Release 6.1.1	Default Router Preferences and More-Specific Routes (RFC 4191) was introduced.
	IPv6 Router Advertisement Options for DNS Configuration (RFC 6106) was introduced.

Prerequisites for Implementing Network Stack IPv4 and IPv6

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing Network Stack IPv4 and IPv6

In any Cisco IOS XR software release with IPv6 support, multiple IPv6 global addresses can be configured on an interface. However, multiple IPv6 link-local addresses on an interface are not supported.

Information About Implementing Network Stack IPv4 and IPv6

To implement Network Stack IPv4 and IPv6, you need to understand the following concepts:

Network Stack IPv4 and IPv6 Exceptions

The Network Stack feature in the Cisco IOS XR software has the following exceptions:

- In Cisco IOS XR software, the **clear ipv6 neighbors** and **show ipv6 neighbors** commands include the **location** *node-id* keyword. If a location is specified, only the neighbor entries in the specified location are displayed.

- The **ipv6 nd scavenge-timeout** command sets the lifetime for neighbor entries in the stale state. When the scavenge-timer for a neighbor entry expires, the entry is cleared.

- In Cisco IOS XR software, the **show ipv4 interface** and **show ipv6 interface** commands include the **location** *node-id* keyword. If a location is specified, only the interface entries in the specified location are displayed.

- Cisco IOS XR software allows conflicting IP address entries at the time of configuration. If an IP address conflict exists between two interfaces that are active, Cisco IOS XR software brings down the interface according to the configured conflict policy, the default policy being to bring down the higher interface instance. For example, if GigabitEthernet 0/1/0/1 conflicts with GigabitEthernet 0/2/0/1, then the IPv4 protocol on GigabitEthernet 0/2/0/1 is brought down and IPv4 remains active on GigabitEthernet 0/1/0/1.

IPv4 and IPv6 Functionality

When Cisco IOS XR software is configured with both an IPv4 and an IPv6 address, the interface can send and receive data on both IPv4 and IPv6 networks.

The architecture of IPv6 has been designed to allow existing IPv4 users to make the transition easily to IPv6 while providing services such as end-to-end security, quality of service (QoS), and globally unique addresses. The larger IPv6 address space allows networks to scale and provide global reachability. The simplified IPv6 packet header format handles packets more efficiently. IPv6 prefix aggregation, simplified network renumbering, and IPv6 site multihoming capabilities provide an IPv6 addressing hierarchy that allows for more efficient routing. IPv6 supports widely deployed routing protocols such as Open Shortest Path First (OSPF), and multiprotocol Border Gateway Protocol (BGP).

The IPv6 neighbor discovery (nd) process uses Internet Control Message Protocol (ICMP) messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and keep track of neighboring routers.

IPv6 for Cisco IOS XR Software

IPv6, formerly named IPng (next generation) is the latest version of the Internet Protocol (IP). IP is a packet-based protocol used to exchange data, voice, and video traffic over digital networks. IPv6 was proposed when it became clear that the 32-bit addressing scheme of IP version 4 (IPv4) was inadequate to meet the demands of Internet growth. After extensive discussion, it was decided to base IPng on IP but add a much larger address space and improvements such as a simplified main header and extension headers. IPv6 is described initially in RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification* issued by the Internet Engineering Task Force (IETF). Further RFCs describe the architecture and services supported by IPv6.

Larger IPv6 Address Space

The primary motivation for IPv6 is the need to meet the anticipated future demand for globally unique IP addresses. Applications such as mobile Internet-enabled devices (such as personal digital assistants [PDAs], telephones, and cars), home-area networks (HANs), and wireless data services are driving the demand for globally unique IP addresses. IPv6 quadruples the number of network address bits from 32 bits (in IPv4) to 128 bits, which provides more than enough globally unique IP addresses for every networked device on the planet. By being globally unique, IPv6 addresses inherently enable global reachability and end-to-end security for networked devices, functionality that is crucial to the applications and services that are driving the demand for the addresses. Additionally, the flexibility of the IPv6 address space reduces the need for private addresses and the use of Network Address Translation (NAT); therefore, IPv6 enables new application protocols that do not require special processing by border routers at the edge of networks.

IPv6 Address Formats

IPv6 addresses are represented as a series of 16-bit hexadecimal fields separated by colons (:) in the format: x:x:x:x:x:x:x:x. Following are two examples of IPv6 addresses:

2001:0DB8:7654:3210:FEDC:BA98:7654:3210

2001:0DB8:0:0:8:800:200C:417A

It is common for IPv6 addresses to contain successive hexadecimal fields of zeros. To make IPv6 addresses less cumbersome, two colons (::) can be used to compress successive hexadecimal fields of zeros at the beginning, middle, or end of an IPv6 address. (The colons represent successive hexadecimal fields of zeros.) lists compressed IPv6 address formats.

A double colon may be used as part of the *ipv6-address* argument when consecutive 16-bit values are denoted as zero. You can configure multiple IPv6 addresses per interfaces, but only one link-local address.

Note Two colons (::) can be used only once in an IPv6 address to represent the longest successive hexadecimal fields of zeros.

The hexadecimal letters in IPv6 addresses are not case-sensitive.

Table 1: Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:0DB8:800:200C:417A	1080::0DB8:800:200C:417A
Multicast	FF01:0:0:0:0:0:0:101	FF01::101
Loopback	0:0:0:0:0:0:0:1	::1
Unspecified	0:0:0:0:0:0:0:0	::

The loopback address listed in Table 1: Compressed IPv6 Address Formats, on page 4 may be used by a node to send an IPv6 packet to itself. The loopback address in IPv6 functions the same as the loopback address in IPv4 (127.0.0.1).

Note The IPv6 loopback address cannot be assigned to a physical interface. A packet that has the IPv6 loopback address as its source or destination address must remain within the node that created the packet. IPv6 routers do not forward packets that have the IPv6 loopback address as their source or destination address.

The unspecified address listed in Table 1: Compressed IPv6 Address Formats, on page 4 indicates the absence of an IPv6 address. For example, a newly initialized node on an IPv6 network may use the unspecified address as the source address in its packets until it receives its IPv6 address.

Note The IPv6 unspecified address cannot be assigned to an interface. The unspecified IPv6 addresses must not be used as destination addresses in IPv6 packets or the IPv6 routing header.

An IPv6 address prefix, in the format *ipv6-prefix/prefix-length* , can be used to represent bit-wise contiguous blocks of the entire address space. The *ipv6-prefix* argument must be in the form documented in RFC 2373, in which the address is specified in hexadecimal using 16-bit values between colons. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address compose the prefix (the network portion of the address). For example, 2001:0DB8:8086:6502::/32 is a valid IPv6 prefix.

IPv6 Address Type: Unicast

An IPv6 unicast address is an identifier for a single interface, on a single node. A packet that is sent to a unicast address is delivered to the interface identified by that address. Cisco IOS XR software supports the following IPv6 unicast address types:

- Global aggregatable address

- Site-local address (proposal to remove by IETF)

- Link-local address

- IPv4-compatible IPv6 address

Aggregatable Global Address

An aggregatable global address is an IPv6 address from the aggregatable global unicast prefix. The structure of aggregatable global unicast addresses enables strict aggregation of routing prefixes that limits the number of routing table entries in the global routing table. Aggregatable global addresses are used on links that are aggregated upward through organizations, and eventually to the Internet service providers (ISPs).

Aggregatable global IPv6 addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Except for addresses that start with binary 000, all global unicast addresses have a 64-bit interface ID. The current global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). Figure 1: Aggregatable Global Address Format, on page 5 shows the structure of an aggregatable global address.

Figure 1: Aggregatable Global Address Format
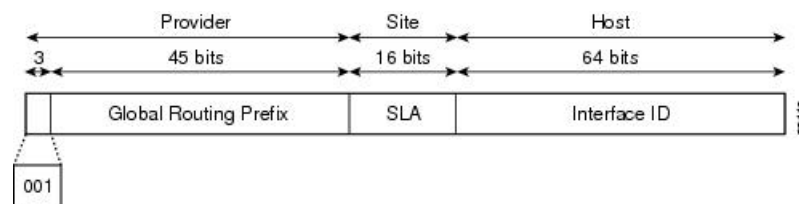
Addresses with a prefix of 2000::/3 (001) through E000::/3 (111) are required to have 64-bit interface identifiers in the extended universal identifier (EUI)-64 format. The Internet Assigned Numbers Authority (IANA) allocates the IPv6 address space in the range of 2000::/16 to regional registries.

The aggregatable global address typically consists of a 48-bit global routing prefix and a 16-bit subnet ID or Site-Level Aggregator (SLA). In the IPv6 aggregatable global unicast address format document (RFC 2374), the global routing prefix included two other hierarchically structured fields named Top-Level Aggregator (TLA) and Next-Level Aggregator (NLA).The IETF decided to remove the TLS and NLA fields from the RFCs, because these fields are policy-based. Some existing IPv6 networks deployed before the change might still be using networks based on the older architecture.

A 16-bit subnet field called the subnet ID could be used by individual organizations to create their own local addressing hierarchy and to identify subnets. A subnet ID is similar to a subnet in IPv4, except that an organization with an IPv6 subnet ID can support up to 65,535 individual subnets.

An interface ID is used to identify interfaces on a link. The interface ID must be unique to the link. It may also be unique over a broader scope. In many cases, an interface ID is the same as or based on the link-layer address of an interface. Interface IDs used in aggregatable global unicast and other IPv6 address types must be 64 bits long and constructed in the modified EUI-64 format.

Interface IDs are constructed in the modified EUI-64 format in one of the following ways:

- For all IEEE 802 interface types (for example, Ethernet interfaces and FDDI interfaces), the first three octets (24 bits) are taken from the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (MAC address) of the interface, the fourth and fifth octets (16 bits) are a fixed hexadecimal value of FFFE, and the last three octets (24 bits) are taken from the last three octets of the MAC address. The construction of the interface ID is completed by setting the Universal/Local (U/L) bit—the seventh bit

of the first octet—to a value of 0 or 1. A value of 0 indicates a locally administered identifier; a value of 1 indicates a globally unique IPv6 interface identifier.

• For all other interface types (for example, serial, loopback, ATM, Frame Relay, and tunnel interface types—except tunnel interfaces used with IPv6 overlay tunnels), the interface ID is constructed in the same way as the interface ID for IEEE 802 interface types; however, the first MAC address from the pool of MAC addresses in the router is used to construct the identifier (because the interface does not have a MAC address).

• For tunnel interface types that are used with IPv6 overlay tunnels, the interface ID is the IPv4 address assigned to the tunnel interface with all zeros in the high-order 32 bits of the identifier.

> **Note** For interfaces using Point-to-Point Protocol (PPP), given that the interfaces at both ends of the connection might have the same MAC address, the interface identifiers used at both ends of the connection are negotiated (picked randomly and, if necessary, reconstructed) until both identifiers are unique. The first MAC address in the router is used to construct the identifier for interfaces using PPP.

If no IEEE 802 interface types are in the router, link-local IPv6 addresses are generated on the interfaces in the router in the following sequence:

1. The router is queried for MAC addresses (from the pool of MAC addresses in the router).

2. If no MAC address is available, the serial number of the Route Processor (RP) or line card (LC) is used to form the link-local address.

Link-Local Address

A link-local address is an IPv6 unicast address that can be automatically configured on any interface using the link-local prefix FE80::/10 (1111 1110 10) and the interface identifier in the modified EUI-64 format. Link-local addresses are used in the neighbor discovery protocol and the stateless autoconfiguration process. Nodes on a local link can use link-local addresses to communicate; the nodes do not need site-local or globally unique addresses to communicate. Figure 2: Link-Local Address Format, on page 6 shows the structure of a link-local address.

IPv6 routers must not forward packets that have link-local source or destination addresses to other links.

Figure 2: Link-Local Address Format
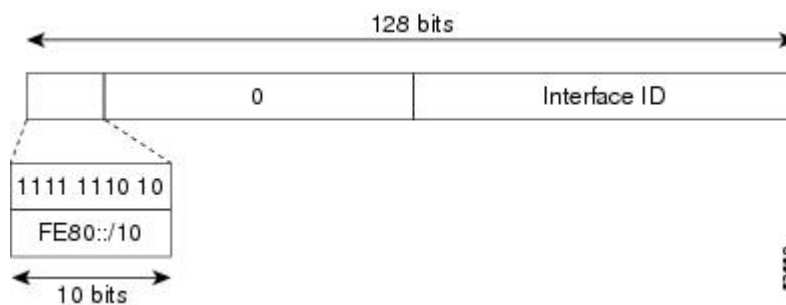
IPv4-Compatible IPv6 Address

An IPv4-compatible IPv6 address is an IPv6 unicast address that has zeros in the high-order 96 bits of the address and an IPv4 address in the low-order 32 bits of the address. The format of an IPv4-compatible IPv6 address is 0:0:0:0:0:0:A.B.C.D or ::A.B.C.D. The entire 128-bit IPv4-compatible IPv6 address is used as the IPv6 address of a node and the IPv4 address embedded in the low-order 32 bits is used as the IPv4 address of the node. IPv4-compatible IPv6 addresses are assigned to nodes that support both the IPv4 and IPv6 protocol stacks and are used in automatic tunnels. Figure 3: IPv4-Compatible IPv6 Address Format, on page 7 shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

Figure 3: IPv4-Compatible IPv6 Address Format
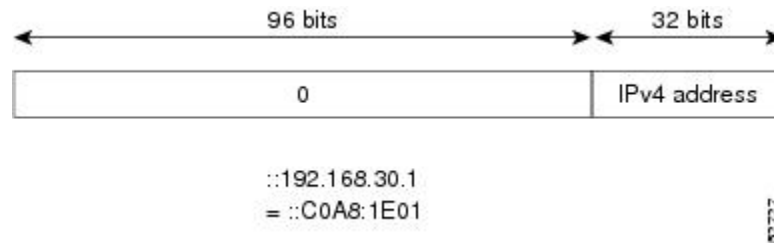

Simplified IPv6 Packet Header

The basic IPv4 packet header has 12 fields with a total size of 20 octets (160 bits). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header. The shaded fields of the IPv4 packet header are not included in the IPv6 packet header. (See Figure 4: IPv4 Packet Header Format, on page 7)

Figure 4: IPv4 Packet Header Format
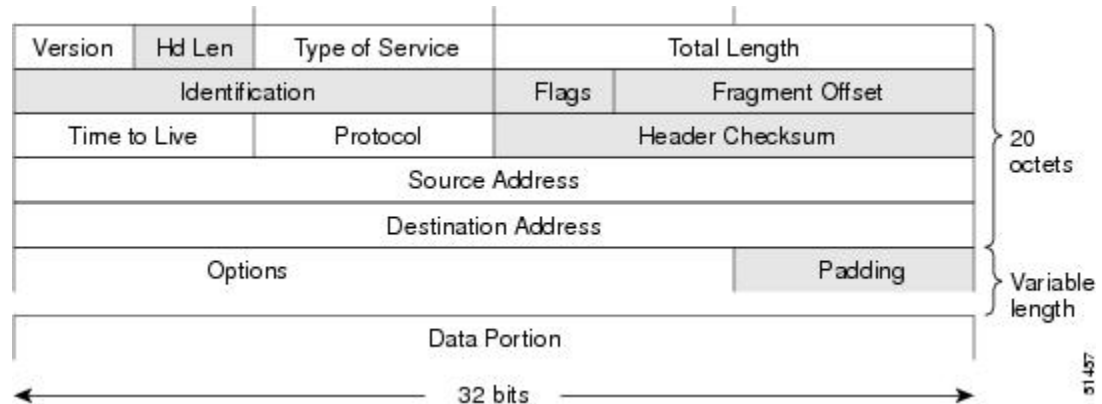
The basic IPv6 packet header has 8 fields with a total size of 40 octets (320 bits). (See Figure 5: IPv6 Packet Header Format, on page 8.) Fields were removed from the IPv6 header because, in IPv6, fragmentation is not handled by routers and checksums at the network layer are not used. Instead, fragmentation in IPv6 is handled by the source of a packet and checksums at the data link layer and transport layer are used. (In IPv4, the User Datagram Protocol (UDP) transport layer uses an optional checksum. In IPv6, use of the UDP checksum is required to check the integrity of the inner packet.) Additionally, the basic IPv6 packet header and Options field are aligned to 64 bits, which can facilitate the processing of IPv6 packets.

Figure 5: IPv6 Packet Header Format
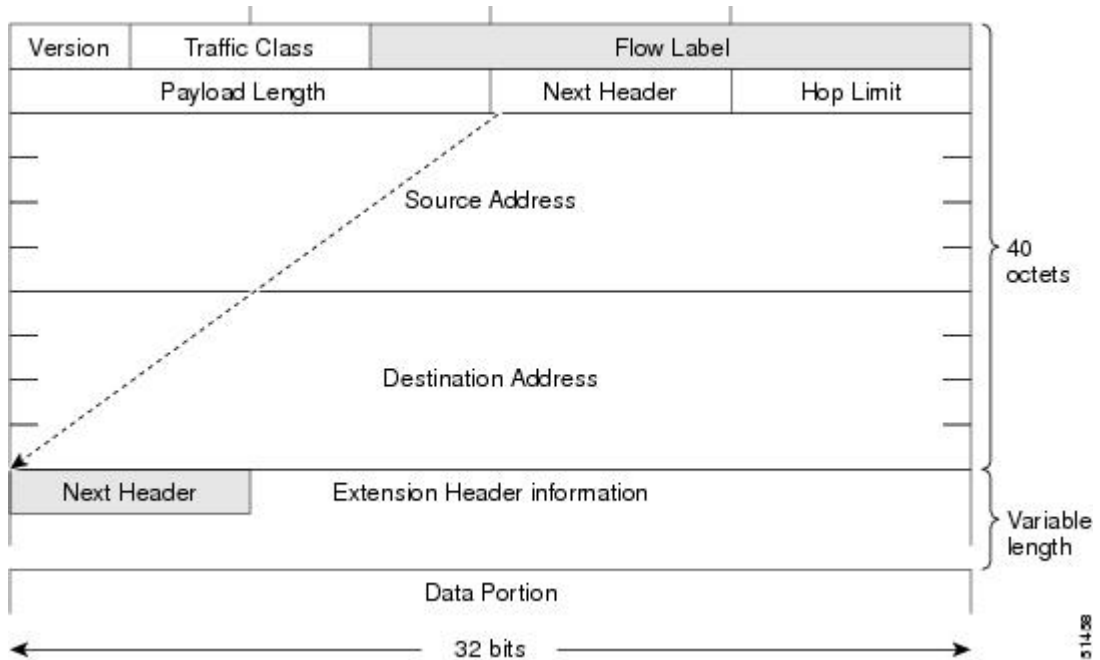
This table lists the fields in the basic IPv6 packet header.

Table 2: Basic IPv6 Packet Header Fields

Field	Description
Version	Similar to the Version field in the IPv4 packet header, except that the field lists number 6 for IPv6 instead of number 4 for IPv4.
Traffic Class	Similar to the Type of Service field in the IPv4 packet header. The Traffic Class field tags packets with a traffic class that is used in differentiated services.
Flow Label	A new field in the IPv6 packet header. The Flow Label field tags packets with a specific flow that differentiates the packets at the network layer.
Payload Length	Similar to the Total Length field in the IPv4 packet header. The Payload Length field indicates the total length of the data portion of the packet.
Next Header	Similar to the Protocol field in the IPv4 packet header. The value of the Next Header field determines the type of information following the basic IPv6 header. The type of information following the basic IPv6 header can be a transport-layer packet, for example, a TCP or UDP packet, or an Extension Header, as shown in Figure 6: IPv6 Extension Header Format, on page 9.
Hop Limit	Similar to the Time to Live field in the IPv4 packet header. The value of the Hop Limit field specifies the maximum number of routers that an IPv6 packet can pass through before the packet is considered invalid. Each router decrements the value by one. Because no checksum is in the IPv6 header, the router can decrement the value without needing to recalculate the checksum, which saves processing resources.

Field	Description
Source Address	Similar to the Source Address field in the IPv4 packet header, except that the field contains a 128-bit source address for IPv6 instead of a 32-bit source address for IPv4.
Destination Address	Similar to the Destination Address field in the IPv4 packet header, except that the field contains a 128-bit destination address for IPv6 instead of a 32-bit destination address for IPv4.

Following the eight fields of the basic IPv6 packet header are optional extension headers and the data portion of the packet. If present, each extension header is aligned to 64 bits. There is no fixed number of extension headers in an IPv6 packet. Together, the extension headers form a chain of headers. Each extension header is identified by the Next Header field of the previous header. Typically, the final extension header has a Next Header field of a transport-layer protocol, such as TCP or UDP. Figure 6: IPv6 Extension Header Format, on page 9 shows the IPv6 extension header format.

Figure 6: IPv6 Extension Header Format
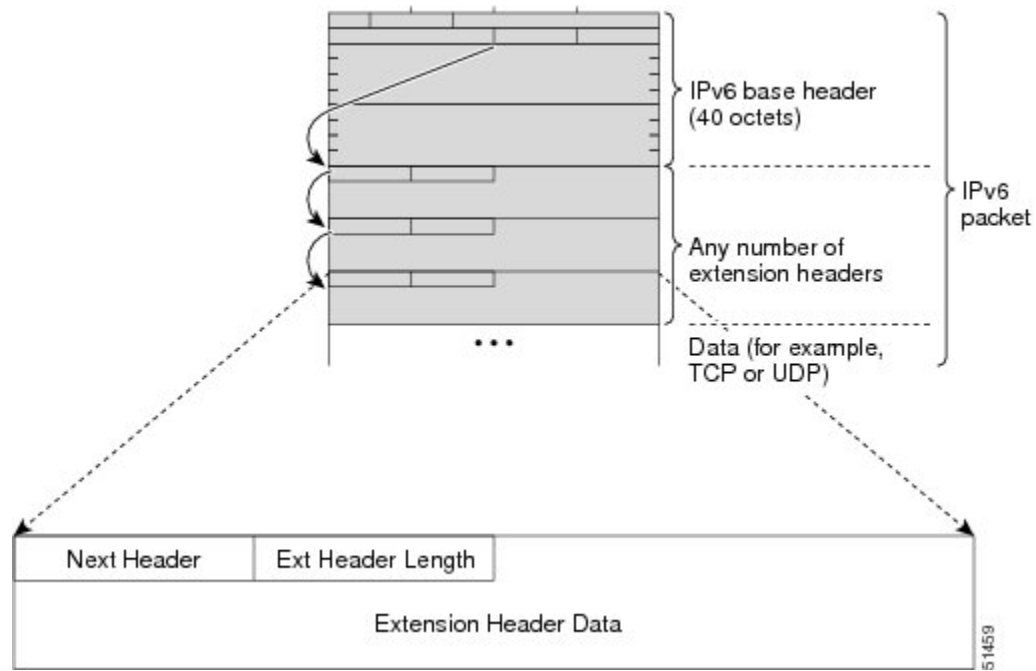
This table lists the extension header types and their Next Header field values.

Table 3: IPv6 Extension Header Types

Header Type	Next Header Value	Description
Hop-by-hop options header	0	This header is processed by all hops in the path of a packet. When present, the hop-by-hop options header always follows immediately after the basic IPv6 packet header.

Header Type	Next Header Value	Description
Destination options header	60	The destination options header can follow any hop-by-hop options header, in which case the destination options header is processed at the final destination and also at each visited address specified by a routing header. Alternatively, the destination options header can follow any Encapsulating Security Payload (ESP) header, in which case the destination options header is processed only at the final destination.
Routing header	43	The routing header is used for source routing.
Fragment header	44	The fragment header is used when a source must fragment a packet that is larger than the maximum transmission unit (MTU) for the path between itself and a destination. The Fragment header is used in each fragmented packet.
Authentication header and ESP header	51 50	The Authentication header and the ESP header are used within IP Security Protocol (IPSec) to provide authentication, integrity, and confidentiality of a packet. These headers are identical for both IPv4 and IPv6.
Upper-layer header	6 (TCP) 17 (UDP)	The upper-layer (transport) headers are the typical headers used inside a packet to transport the data. The two main transport protocols are TCP and UDP.
Mobility header	To be done by IANA	Extension headers used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings.

Path MTU Discovery for IPv6

As in IPv4, path MTU discovery in IPv6 allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path. In IPv6, however, fragmentation is handled by the source of a packet when the path MTU of one link along a given data path is not large enough to accommodate the size of the packets. Having IPv6 hosts handle packet fragmentation saves IPv6 router processing resources and helps IPv6 networks run more efficiently.

In IPv4, the minimum link MTU is 68 octets, which means that the MTU size of every link along a given data path must support an MTU size of at least 68 octets. In IPv6, the minimum link MTU is 1280 octets. We recommend using an MTU value of 1500 octets for IPv6 links.

Note Path MTU discovery is supported only for applications using TCP and Ping protocol.

To enable path MTU discovery in IPv6 for applications using the Ping protocol, the path MTU command must be enabled. To do so, run the following command in global configuration mode:

```
RP/0/RSP0/CPU0:router(config)# ipv6 path-mtu enable
```
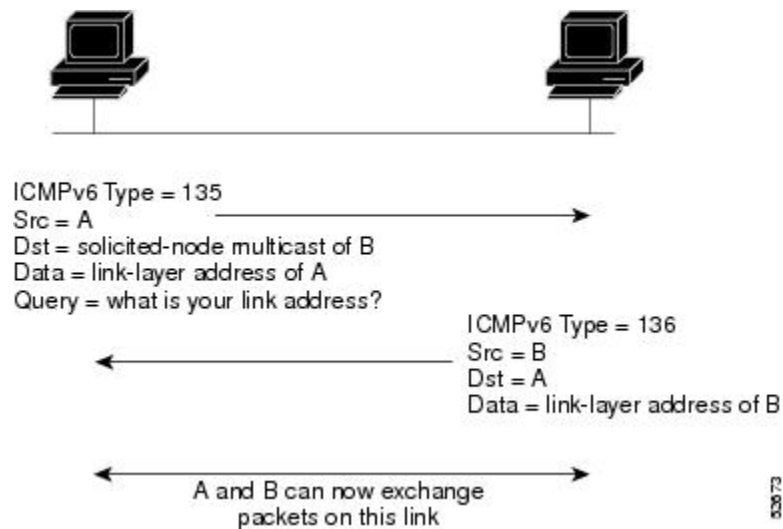
IPv6 Neighbor Discovery

The IPv6 neighbor discovery process uses ICMP messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and keep track of neighboring routers.

IPv6 Neighbor Solicitation Message

A value of 135 in the Type field of the ICMP packet header identifies a neighbor solicitation message. Neighbor solicitation messages are sent on the local link when a node wants to determine the link-layer address of another node on the same local link. (See Figure 7: IPv6 Neighbor Discovery—Neighbor Solicitation Message, on page 11.) When a node wants to determine the link-layer address of another node, the source address in a neighbor solicitation message is the IPv6 address of the node sending the neighbor solicitation message. The destination address in the neighbor solicitation message is the solicited-node multicast address that corresponds to the IPv6 address of the destination node. The neighbor solicitation message also includes the link-layer address of the source node.

Figure 7: IPv6 Neighbor Discovery—Neighbor Solicitation Message

ICMPv6 Type = 135
Src = A
Dst = solicited-node multicast of B
Data = link-layer address of A
Query = what is your link address?

ICMPv6 Type = 136
Src = B
Dst = A
Data = link-layer address of B

A and B can now exchange
packets on this link

After receiving the neighbor solicitation message, the destination node replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header, on the local link. The source address in the neighbor advertisement message is the IPv6 address of the node (more specifically, the IPv6 address of the node interface) sending the neighbor advertisement message. The destination address in the neighbor advertisement message is the IPv6 address of the node that sent the neighbor solicitation message. The data portion of the neighbor advertisement message includes the link-layer address of the node sending the neighbor advertisement message.

After the source node receives the neighbor advertisement, the source node and destination node can communicate.

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. When a node wants to verifying the reachability of a neighbor, the destination address in a neighbor solicitation message is the unicast address of the neighbor.

Neighbor advertisement messages are also sent when there is a change in the link-layer address of a node on a local link. When there is such a change, the destination address for the neighbor advertisement is the all-nodes multicast address.

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. Neighbor unreachability detection identifies the failure of a neighbor or the failure of the forward path to the neighbor, and is used for all paths between hosts and neighboring nodes (hosts or routers). Neighbor unreachability detection is performed for neighbors to which only unicast packets are being sent and is not performed for neighbors to which multicast packets are being sent.

A neighbor is considered reachable when a positive acknowledgment is returned from the neighbor (indicating that packets previously sent to the neighbor have been received and processed). A positive acknowledgment—from an upper-layer protocol (such as TCP)—indicates that a connection is making forward progress (reaching its destination) or that a neighbor advertisement message in response to a neighbor solicitation message has been received. If packets are reaching the peer, they are also reaching the next-hop neighbor of the source. Therefore, forward progress is also a confirmation that the next-hop neighbor is reachable.

For destinations that are not on the local link, forward progress implies that the first-hop router is reachable. When acknowledgments from an upper-layer protocol are not available, a node probes the neighbor using unicast neighbor solicitation messages to verify that the forward path is still working. The return of a solicited neighbor advertisement message from the neighbor is a positive acknowledgment that the forward path is still working. (Neighbor advertisement messages that have the solicited flag set to a value of 1 are sent only in response to a neighbor solicitation message.) Unsolicited messages confirm only the one-way path from the source to the destination node; solicited neighbor advertisement messages indicate that a path is working in both directions.

Note   A neighbor advertisement message that has the solicited flag set to a value of 0 must not be considered as a positive acknowledgment that the forward path is still working.

Neighbor solicitation messages are also used in the stateless autoconfiguration process to verify the uniqueness of unicast IPv6 addresses before the addresses are assigned to an interface. Duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned to an interface. (The new address remains in a tentative state while duplicate address detection is performed.) Specifically, a node sends a neighbor solicitation message with an unspecified source address and a tentative link-local address in the body of the message. If another node is already using that address, the node returns a neighbor advertisement message that contains the tentative link-local address. If another node is simultaneously verifying the uniqueness of the same address, that node also returns a neighbor solicitation message. If no neighbor advertisement messages are received in response to the neighbor solicitation message and no neighbor solicitation messages are received from other nodes that are attempting to verify the same tentative address, the node that sent the original neighbor solicitation message considers the tentative link-local address to be unique and assigns the address to the interface.

Every IPv6 unicast address (global or link-local) must be checked for uniqueness on the link; however, until the uniqueness of the link-local address is verified, duplicate address detection is not performed on any other IPv6 addresses associated with the link-local address. The Cisco implementation of duplicate address detection in the Cisco IOS XR software does not check the uniqueness of anycast or global addresses that are generated from 64-bit interface identifiers.

IPv6 Router Advertisement Message

Router advertisement (RA) messages, which have a value of 134 in the Type field of the ICMP packet header, are periodically sent out each configured interface of an IPv6 router. The router advertisement messages are sent to the all-nodes multicast address. (See Figure 8: IPv6 Neighbor Discovery—Router Advertisement Message, on page 13.)

Figure 8: IPv6 Neighbor Discovery—Router Advertisement Message
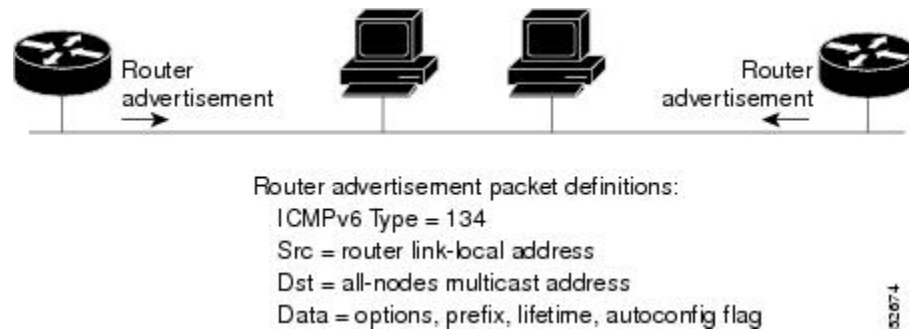
Router advertisement messages typically include the following information:

- One or more onlink IPv6 prefixes that nodes on the local link can use to automatically configure their IPv6 addresses

- Lifetime information for each prefix included in the advertisement

- Sets of flags that indicate the type of autoconfiguration (stateless or statefull) that can be completed

- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time, in seconds, that the router should be used as a default router)

- Additional information for hosts, such as the hop limit and MTU a host should use in packets that it originates

Router advertisements are also sent in response to router solicitation messages. Router solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled router advertisement message. Given that router solicitation messages are usually sent by hosts at system startup (the host does not have a configured unicast address), the source address in router solicitation messages is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the unicast address of the interface sending the router solicitation message is used as the source address in the message. The destination address in router solicitation messages is the all-routers multicast address with a scope of the link. When a router advertisement is sent in response to a router solicitation, the destination address in the router advertisement message is the unicast address of the source of the router solicitation message.

The following router advertisement message parameters can be configured:

- The time interval between periodic router advertisement messages

- The "router lifetime" value, which indicates the usefulness of a router as the default router (for use by all nodes on a given link)

- The network prefixes in use on a given link

- The time interval between neighbor solicitation message retransmissions (on a given link)

• The amount of time a node considers a neighbor reachable (for use by all nodes on a given link)

The configured parameters are specific to an interface. The sending of router advertisement messages (with default values) is automatically enabled on Ethernet and FDDI interfaces. For other interface types, the sending of router advertisement messages must be manually configured by using the **no ipv6 nd suppress-ra** command in interface configuration mode. The sending of router advertisement messages can be disabled on individual interfaces by using the **ipv6 nd suppress-ra** command in interface configuration mode.

Note   For stateless autoconfiguration to work properly, the advertised prefix length in router advertisement messages must always be 64 bits.

IPv6 Neighbor Redirect Message

A value of 137 in the Type field of the ICMP packet header identifies an IPv6 neighbor redirect message. Routers send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination. (See Figure 9: IPv6 Neighbor Discovery—Neighbor Redirect Message , on page 14.)

Figure 9: IPv6 Neighbor Discovery—Neighbor Redirect Message
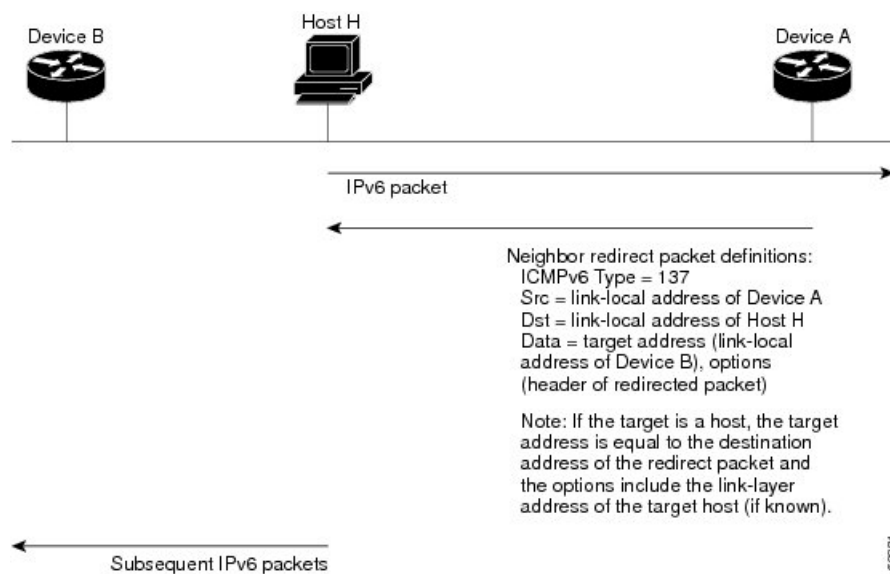
Note   A router must be able to determine the link-local address for each of its neighboring routers to ensure that the target address (the final destination) in a redirect message identifies the neighbor router by its link-local address. For static routing, the address of the next-hop router should be specified using the link-local address of the router; for dynamic routing, all IPv6 routing protocols must exchange the link-local addresses of neighboring routers.

After forwarding a packet, a router should send a redirect message to the source of the packet under the following circumstances:

• The destination address of the packet is not a multicast address.

> • The packet was not addressed to the router.
>
> • The packet is about to be sent out the interface on which it was received.
>
> • The router determines that a better first-hop node for the packet resides on the same link as the source of the packet.
>
> • The source address of the packet is a global IPv6 address of a neighbor on the same link, or a link-local address.

Use the **ipv6 icmp error-interval** global configuration command to limit the rate at which the router generates all IPv6 ICMP error messages, including neighbor redirect messages, which ultimately reduces link-layer congestion.

Note   A router must not update its routing tables after receiving a neighbor redirect message, and hosts must not originate neighbor redirect messages.

ICMP for IPv6

Internet Control Message Protocol (ICMP) in IPv6 functions the same as ICMP in IPv4—ICMP generates error messages, such as ICMP destination unreachable messages and informational messages like ICMP echo request and reply messages. Additionally, ICMP packets in IPv6 are used in the IPv6 neighbor discovery process, path MTU discovery, and the Multicast Listener Discovery (MLD) protocol for IPv6. MLD is used by IPv6 routers to discover multicast listeners (nodes that want to receive multicast packets destined for specific multicast addresses) on directly attached links. MLD is based on version 2 of the Internet Group Management Protocol (IGMP) for IPv4.

A value of 58 in the Next Header field of the basic IPv6 packet header identifies an IPv6 ICMP packet. ICMP packets in IPv6 are like a transport-layer packet in the sense that the ICMP packet follows all the extension headers and is the last piece of information in the IPv6 packet. Within IPv6 ICMP packets, the ICMPv6 Type and ICMPv6 Code fields identify IPv6 ICMP packet specifics, such as the ICMP message type. The value in the Checksum field is derived (computed by the sender and checked by the receiver) from the fields in the IPv6 ICMP packet and the IPv6 pseudoheader. The ICMPv6 Data field contains error or diagnostic information relevant to IP packet processing.

Address Repository Manager

IPv4 and IPv6 Address Repository Manager (IPARM) enforces the uniqueness of global IP addresses configured in the system, and provides global IP address information dissemination to processes on route processors (RPs) and line cards (LCs) using the IP address consumer application program interfaces (APIs), which includes unnumbered interface information.

Address Conflict Resolution

There are two parts to conflict resolution; the conflict database and the conflict set definition.

Conflict Database

IPARM maintains a global conflict database. IP addresses that conflict with each other are maintained in lists called conflict sets. These conflict sets make up the global conflict database.

A set of IP addresses are said to be part of a conflict set if at least one prefix in the set conflicts with every other IP address belonging to the same set. For example, the following four addresses are part of a single conflict set.

address 1: 10.1.1.1/16

address 2: 10.2.1.1/16

address 3: 10.3.1.1/16

address 4: 10.4.1.1/8

When a conflicting IP address is added to a conflict set, an algorithm runs through the set to determine the highest precedence address within the set.

This conflict policy algorithm is deterministic, that is, the user can tell which addresses on the interface are enabled or disabled. The address on the interface that is enabled is declared as the highest precedence ip address for that conflict set.

The conflict policy algorithm determines the highest precedence ip address within the set.

Multiple IP Addresses

The IPARM conflict handling algorithm allows multiple IP addresses to be enabled within a set. Multiple addresses could potentially be highest precedence IP addresses.

interface GigabitEthernet 0/2/0/0: 10.1.1.1/16

interface GigabitEthernet 0/3/0/0: 10.1.1.2/8

interface GigabitEthernet 0/4/0/0: 10.2.1.1/16

The IP address on GigabitEthernet 0/2/0/0 is declared as highest precedence as per the lowest rack/slot policy and is enabled. However, because the address on interface GigabitEthernet 0/4/0/0 does not conflict with the current highest precedence IP address, the address on GigabitEthernet 0/4/0/0 is enabled as well.

Recursive Resolution of Conflict Sets

In the example below, the address on the interface in GigabitEthernet 0/2/0/0 has the highest precedence because it is the lowest rack/slot. However, now the addresses on GigabitEthernet 0/4/0/0 and GigabitEthernet 0/5/0/0 also do not conflict with the highest precedence IP addresses on GigabitEthernet 0/2/0/0. However, the addresses on GigabitEthernet 0/4/0/0 and GigabitEthernet 0/5/0/0 conflict with each other. The conflict resolution software tries to keep the interface that is enabled as the one that needs to stay enabled. If both interfaces are disabled, the software enables the address based on the current conflict policy. Because GigabitEthernet 0/4/0/0 is on a lower rack/slot, it is enabled.

interface GigabitEthernet 0/2/0/0: 10.1.1.1/16

interface GigabitEthernet 0/3/0/0: 10.1.1.2/8

interface GigabitEthernet 0/4/0/0: 10.2.1.1/16

interface GigabitEthernet 0/5/0/0: 10.2.1.2/16

Route-Tag Support for Connected Routes

The Route-Tag Support for Connected Routes feature that attaches a tag with all IPv4 and IPv6 addresses of an interface. The tag is propagated from the IPv4 and IPv6 management agents (MA) to the IPv4 and IPv6 address repository managers (ARM) to routing protocols, thus enabling the user to control the redistribution

of connected routes by looking at the route tags, by using routing policy language (RPL) scripts. This prevents the redistribution of some interfaces, by checking for route tags in a route policy.

The route tag feature is already available for static routes and connected routes (interfaces) wherein the route tags are matched to policies and redistribution can be prevented.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. Do one of the following:

    • **ipv4 address** *ipv4-address mask* [**secondary**]

4. route-tag *[route-tag value]*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1`	Enters interface configuration mode.
Step 3	Do one of the following: • **ipv4 address** *ipv4-address mask* [**secondary**] **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.0.0.0`	Specifies a primary (or secondary) IPv4 address address for an interface.

Step 4	route-tag *[route-tag value]*
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.0.0.0`
	`route-tag`
	` 100`
		Specifies that the configured address has a route tag to be associated with it. The range for the route-tag value is 1 to 4294967295.

Step 5	**commit**	

How to Implement Network Stack IPv4 and IPv6

This section contains the following procedures:

Assigning IPv4 Addresses to Network Interfaces

This task assigns IPv4 addresses to individual network interfaces.

IPv4 Addresses

A basic and required task for configuring IP is to assign IPv4 addresses to network interfaces. Doing so enables the interfaces and allows communication with hosts on those interfaces using IPv4. An IP address identifies a location to which IP datagrams can be sent. An interface can have one primary IP address and multiple (up to 500) secondary addresses. Packets generated by the software always use the primary IPv4 address. Therefore, all networking devices on a segment should share the same primary network number.

Associated with this task are decisions about subnetting and masking the IP addresses. A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a *subnet mask*.

Note   Cisco supports only network masks that use contiguous bits that are flush left against the network field.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ipv4-address mask* [**secondary**]
4. **commit**
5. show ipv4 interface

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1`	Enters interface configuration mode.

Step 3	**ipv4 address** *ipv4-address mask* [**secondary**]
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.0.0.0`
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.27/8`
		Specifies a primary or secondary IPv4 address for an interface.
		
		• The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means the corresponding address bit belongs to the network address.
		
		• The network mask can be indicated as a slash (/) and a number- a prefix length. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash

	Command or Action	Purpose
		must precede the decimal value, and there is no space between the IP address and the slash.
Step 4	**commit**	
Step 5	show ipv4 interface **Example:** `RP/0/RSP0/CPU0:router# show ipv4 interface`	(Optional) Displays the usability status of interfaces configured for IPv4.

IPv4 Virtual Addresses

Configuring an IPv4 virtual address enables you to access the router from a single virtual address with a management network, without the prior knowledge of which route processor (RP) is active. An IPv4 virtual address persists across RP failover situations. For this to happen, the virtual IPv4 address must share a common IPv4 subnet with a Management Ethernet interface on both RPs.

The **vrf** keyword supports virtual addresses on a per-VRF basis.

The **use-as-src-addr** keyword eliminates the need for configuring a loopback interface as the source interface (that is, update source) for management applications. When an update source is not configured, management applications allow the transport processes (TCP, UDP, raw_ip) to select a suitable source address. The transport processes, in turn, consult the FIB for selecting a suitable source address. If a Management Ethernet's IP address is selected as the source address and if the **use-as-src-addr** keyword is configured, then the transport substitutes the Management Ethernet's IP address with a relevant virtual IP address. This functionality works across RP switchovers. If the **use-as-src-addr** is not configured, then the source-address selected by transports can change after a failover and the NMS software may not be able to manage this situation.

Note Protocol configuration such as tacacs source-interface, snmp-server trap-source, ntp source, logging source-interface do not use the virtual management IP address as their source by default. Use the **ipv4 virtual address use-as-src-addr** command to ensure that the protocol uses the virtual IPv4 address as its source address. Alternatively, you can also configure a loopback address with the designated or desired IPv4 address and set that as the source for protocols such as TACACS+ via the **tacacs source-interface** command.

Configuring IPv6 Addressing

This task assigns IPv6 addresses to individual router interfaces and enable the forwarding of IPv6 traffic globally on the router. By default, IPv6 addresses are not configured.

Note The *ipv6-prefix* argument in the **ipv6 address** command must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons.

The **/prefix-length** argument in the **ipv6 address** command is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address) A slash must precede the decimal value.

The *ipv6-address* argument in the **ipv6 address link-local** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

IPv6 Virtual Addresses

Configuring an IPv6 virtual address enables you to access the router from a single virtual address with a management network, without the prior knowledge of which route processor (RP) is active. An IPv6 virtual address persists across RP failover situations. For this to happen, the virtual IPv6 address must share a common IPv6 subnet with a Management Ethernet interface on both RPs.

The **vrf** keyword supports virtual addresses on a per-VRF basis.

The **use-as-src-addr** keyword eliminates the need for configuring a loopback interface as the source interface (that is, update source) for management applications. When an update source is not configured, management applications allow the transport processes (TCP, UDP, raw_ip) to select a suitable source address. The transport processes, in turn, consult the FIB for selecting a suitable source address. If a Management Ethernet's IP address is selected as the source address and if the **use-as-src-addr** keyword is configured, then the transport substitutes the Management Ethernet's IP address with a relevant virtual IP address. This functionality works across RP switchovers. If the **use-as-src-addr** is not configured, then the source-address selected by transports can change after a failover and the NMS software may not be able to manage this situation.

Note Protocol configuration such as tacacs source-interface, snmp-server trap-source, ntp source, logging source-interface do not use the virtual management IP address as their source by default. Use the **ipv6 virtual address use-as-src-addr** command to ensure that the protocol uses the virtual IPv6 address as its source address. Alternatively, you can also configure a loopback address with the designated or desired IPv6 address and set that as the source for protocols such as TACACS+ via the **tacacs source-interface** command.

Assigning Multiple IP Addresses to Network Interfaces

This task assigns multiple IP addresses to network interfaces.

Secondary IPv4 Addresses

The Cisco IOS XR software supports multiple IP addresses per interface.

You can specify a maximum of 500 secondary addresses. Secondary IP addresses can be used in a variety of situations. The following are the most common applications:

- There might not be enough host addresses for a particular network segment. For example, suppose your subnetting allows up to 254 hosts per logical subnet, but on one physical subnet you must have 300 host addresses. Using secondary IP addresses on the routers or access servers allows you to have two logical subnets using one physical subnet.

- Many older networks were built using Level 2 bridges, and were not subnetted. The judicious use of secondary addresses can aid in the transition to a subnetted, router-based network. Routers on an older, bridged segment can easily be made aware that many subnets are on that segment.

- Two subnets of a single network might otherwise be separated by another network. You can create a single network from subnets that are physically separated by another network by using a secondary address. In these instances, the first network is *extended*, or layered on top of the second network. Note that a subnet cannot appear on more than one active interface of the router at a time.

Note	If any router on a network segment uses a secondary IPv4 address, all other routers on that same segment must also use a secondary address from the same network or subnet.

Caution	Inconsistent use of secondary addresses on a network segment can quickly cause routing loops.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ipv4-address mask* [**secondary**]
4. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**interface** *type interface-path-id*
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# interface`
	`gigabitethernet 0/1/0/3`
		Enters interface configuration mode.

Step 3	**ipv4 address** *ipv4-address mask* [**secondary**]
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address`
	`192.168.1.27 255.255.255.0 secondary`
		Specifies that the configured address is a secondary IPv4 address.

Step 4	**commit**	

Configuring IPv4 and IPv6 Protocol Stacks

This task configures an interface in a Cisco networking device to support both the IPv4 and IPv6 protocol stacks.

When an interface in a Cisco networking device is configured with both an IPv4 and an IPv6 address, the interface forwards both IPv4 and IPv6 traffic—the interface can send and receive data on both IPv4 and IPv6 networks.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ip-address mask* [**secondary**]

4. **ipv6 address** *ipv6-prefix*/*prefix-length* [**eui-64**]
5. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**interface** *type interface-path-id*
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config)# interface`
	`gigabitethernet 0/1/0/1`
		Specifies the interface type and number, and enters interface configuration mode.

Step 3	**ipv4 address** *ip-address mask* [**secondary**]
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config-if)# ipv4 address`
	`192.168.99.1 255.255.255.0`
		Specifies a primary or secondary IPv4 address for an interface.

Step 4	**ipv6 address** *ipv6-prefix*/*prefix-length* [**eui-64**]
	
	Example:
	
	`RP/0/RSP0/CPU0:router(config-if)# ipv6 address`
	`2001:0DB8:c18:1::3/64`
		Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface.
		
		• A slash mark (/) must precede the *prefix-length* , and there is no space between the *ipv6-prefix* and slash mark.

Step 5	**commit**	

Enabling IPv4 Processing on an Unnumbered Interface

This task enables IPv4 processing on an unnumbered interface.

IPv4 Processing on an Unnumbered Interface

This section describes the process of enabling an IPv4 point-to-point interface without assigning an explicit IP address to the interface. Whenever the unnumbered interface generates a packet (for example, for a routing update), it uses the address of the interface you specified as the source address of the IP packet. It also uses the specified interface address in determining which routing processes are sending updates over the unnumbered interface. Restrictions are as follows:

- Serial interfaces using High-Level Data Link Control (HDLC), PPP, and Frame Relay encapsulations can be unnumbered. Serial interfaces using Frame Relay encapsulation can also be unnumbered, but the interface must be a point-to-point subinterface.

- You cannot use the **ping** EXEC command to determine whether the interface is up, because the interface has no IP address. The Simple Network Management Protocol (SNMP) can be used to remotely monitor interface status.

- You cannot support IP security options on an unnumbered interface.

- If you have configured GRE tunnel as IPv4 unnumberered interface then you must also configure a static route. The tunnel cannot reach the peer address if there is no static route configured. Here is the configuration example:

```
interface Loopback 100
ipv4 address 192.0.2.1 255.255.255.252

interface tunnel-ip 100
ipv4 unnumbered Loopback 100
tunnel source 192.0.2.10
keepalive
tunnel destination 192.0.2.11

router static
address-family ipv4 unicast
  192.0.2.2/32 tunnel-ip 100
```

If you are configuring Intermediate System-to-Intermediate System (IS-IS) across a serial line, you should configure the serial interfaces as unnumbered, which allows you to conform with RFC 1195, which states that IP addresses are not required on each interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 unnumbered** *interface-type* *interface-instance*
4. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1`	Enters interface configuration mode.
Step 3	**ipv4 unnumbered** *interface-type* *interface-instance* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 unnumbered loopback 5`	Enables IPv4 processing on a point-to-point interface without assigning an explicit IPv4 address to that interface. • The interface you specify must be the name of another interface in the router that has an IP address, not another unnumbered interface. • The interface you specify by the *interface-type* and *interface-instance* arguments must be enabled (listed as "up" in the **show interfaces** command display).
Step 4	**commit**	

Configuring ICMP Rate Limiting

This task explains how to configure IPv4 or IPv6 ICMP rate limiting.

IPv4 ICMP Rate Limiting

The IPv4 ICMP rate limiting feature limits the rate that IPv4 ICMP destination unreachable messages are generated. The Cisco IOS XR software maintains two timers: one for general destination unreachable messages and one for DF destination unreachable messages. Both share the same time limits and defaults. If the **DF** keyword is not configured, the **icmp ipv4 rate-limit unreachable** command sets the time values for DF destination unreachable messages. If the **DF** keyword is configured, its time values remain independent from those of general destination unreachable messages.

IPv6 ICMP Rate Limiting

The IPv6 ICMP rate limiting feature implements a token bucket algorithm for limiting the rate at which IPv6 ICMP error messages are sent out on the network. The initial implementation of IPv6 ICMP rate limiting defined a fixed interval between error messages, but some applications, such as traceroute, often require replies to a group of requests sent in rapid succession. The fixed interval between error messages is not flexible enough to work with applications such as traceroute and can cause the application to fail. Implementing a token bucket scheme allows a number of tokens—representing the ability to send one error message each—to be stored in a virtual bucket. The maximum number of tokens allowed in the bucket can be specified, and for every error message to be sent, one token is removed from the bucket. If a series of error messages is generated, error messages can be sent until the bucket is empty. When the bucket is empty of tokens, IPv6 ICMP error messages are not sent until a new token is placed in the bucket. The token bucket algorithm does not increase the average rate limiting time interval, and it is more flexible than the fixed time interval scheme.

SUMMARY STEPS

1. **configure**
2. Do one of the following:

   - **icmp ipv4 rate-limit unreachable** [**DF**] *milliseconds*
   - **ipv6 icmp error-interval** *milliseconds* [*bucketsize*]

3. **commit**
4. Do one of the following:

   - **show ipv4 traffic** [**brief**]
   - **show ipv6 traffic** [**brief**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	Do one of the following: • **icmp ipv4 rate-limit unreachable** [**DF**] *milliseconds* • **ipv6 icmp error-interval** *milliseconds* [*bucketsize*] **Example:**	Limits the rate that IPv4 ICMP destination unreachable messages are generated. • The **DF** keyword limits the rate at which ICMP destination unreachable messages are sent when code 4 fragmentation is needed and Data Fragmentation

	Command or Action	Purpose

	`RP/0/RSP0/CPU0:router(config)# icmp ipv4 rate-limit unreachable 1000`
	
	or
	
	`RP/0/RSP0/CPU0:router(config)# ipv6 icmp error-interval 50 20`
		(DF) is set, as specified in the IP header of the ICMP destination unreachable message.
		
		• The *milliseconds* argument specifies the time period between the sending of ICMP destination unreachable messages.
		
		or
		
		Configures the interval and bucket size for IPv6 ICMP error messages.
		
		• The *milliseconds* argument specifies the interval between tokens being added to the bucket.
		
		• The optional *bucketsize* argument defines the maximum number of tokens stored in the bucket.

Step 3	**commit**	

Step 4	Do one of the following:
	• **show ipv4 traffic** [**brief**]
	• **show ipv6 traffic** [**brief**]
	Example:
	
	`RP/0/RSP0/CPU0:router# show ipv4 traffic`
	or
	
	`RP/0/RSP0/CPU0:router# show ipv6 traffic`
		(Optional) Displays statistics about IPv4 traffic, including ICMP unreachable information.
		
		• Use the **brief** keyword to display only IPv4 and ICMPv4 traffic statistics.
		
		or
		
		(Optional) Displays statistics about IPv6 traffic, including IPv6 ICMP rate-limited counters.
		
		• Use the **brief** keyword to display only IPv6 and ICMPv6 traffic statistics.

Configuring IPARM Conflict Resolution

This task sets the IP Address Repository Manager (IPARM) address conflict resolution parameters.

Static Policy Resolution

The static policy resolution configuration prevents new address configurations from affecting interfaces that are currently running.

Note   When you configure duplicate IP addresses of interfaces on a device and also configure the command ipv4 conflict-policy static, the duplicate interface remains down. However, this configuration is applicable only on ethernet interfaces and not on Point-to-Point (PPP) interfaces and Cisco ASR 9000 Series SPA Interface Processor-700 (SIP-700).

SUMMARY STEPS

1. **configure**
2. **{ipv4 | ipv6} conflict-policy static**
3. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**{ipv4	ipv6} conflict-policy static**
		
		Example:
		
		`RP/0/RSP0/CPU0:router(config)# ipv4 conflict-policy static`
		
		or
		
		`RP/0/RSP0/CPU0:router(config)# ipv6 conflict-policy static`
			Sets the conflict policy to static, that is, prevents new interface addresses from affecting the currently running interface.

Step 3	**commit**	

Longest Prefix Address Conflict Resolution

This conflict resolution policy attempts to give highest precedence to the IP address that has the longest prefix length.

SUMMARY STEPS

1. **configure**
2. **{ ipv4 | ipv6 } conflict-policy longest-prefix**
3. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**{ ipv4	ipv6 } conflict-policy longest-prefix**
		
		Example:
		
		`RP/0/RSP0/CPU0:router(config)# ipv4 conflict-policy longest-prefix`
		
		or
		
		`RP/0/RSP0/CPU0:router(config)# ipv6 conflict-policy longest-prefix`
			Sets the conflict policy to longest prefix, that is, all addresses within the conflict set that don't conflict with the longest prefix address of the currently running interface are allowed to run as well.

Step 3	**commit**	

Highest IP Address Conflict Resolution

This conflict resolution policy attempts to give highest precedence to the IP address that has the highest value.

SUMMARY STEPS

1. **configure**
2. **{ ipv4 | ipv6 } conflict-policy highest-ip**
3. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**{ ipv4	ipv6 } conflict-policy highest-ip**
		
		Example:
		
		`RP/0/RSP0/CPU0:router(config)# ipv4 conflict-policy`
		`highest-ip`
		
		or
		
		`RP/0/RSP0/CPU0:router(config)# ipv6 conflict-policy`
		`highest-ip`
			Sets the conflict policy to the highest IP value, that is, the IP address with the highest value gets precedence.

Step 3	**commit**	

Generic Routing Encapsulation

The Generic Routing Encapsulation (GRE) tunneling protocol provides a simple, and generic approach for transporting packets of one protocol over another protocol by means of encapsulation. The packet that needs to be transported is first encapsulated in a GRE header, which is further encapsulated in another protocol like IPv4 or IPv6; and the packet is then forwarded to the destination.

A typical GRE-encapsulated packet includes:

- The delivery header
- The GRE header
- The payload packet

A payload packet is a packet that a system encapsulates and delivers to a destination. The payload is first encapsulated in a GRE packet. The resulting GRE packet can then be encapsulated in another outer protocol and then forwarded. This outer protocol is called the delivery protocol.

Note

- When IPv4 is being carried as the GRE payload, the Protocol Type field must be set to 0x800.

- When IPv6 is being carried as the GRE payload, the Protocol Type field must be set to 0x86DD.

IPv4/IPv6 Forwarding over GRE Tunnels

Packets that are tunneled over GRE tunnels enter the router as normal IP packets. The packets are forwarded (routed) using the destination address of the IP packet. In the case of Equal Cost Multi Path (ECMP) scenarios, an output interface-adjacency is selected, based on a platform-specific L3 load balance (LB) hash. Once the egress physical interface is known, the packet is sent out of that interface, after it is first encapsulated with GRE header followed by the L2 rewrite header of the physical interface. After the GRE encapsulated packet reaches the remote tunnel endpoint router, the GRE packet is decapsulated. The destination address lookup of the outer IP header (this is the same as the tunnel destination address) will find a local address (receive) entry on the ingress line card.

The first step in GRE decapsulation is to qualify the tunnel endpoint, before admitting the GRE packet into the router, based on the combination of tunnel source (the same as source IP address of outer IP header) and tunnel destination (the same as destination IP address of outer IP header). If the received packet fails tunnel admittance qualification check, the packet is dropped by the decapsulation router. On successful tunnel admittance check, the decapsulation strips the outer IP and GRE header off the packet, then starts processing the inner payload packet as a regular packet.

When a tunnel endpoint decapsulates a GRE packet, which has an IPv4/IPv6 packet as the payload, the destination address in the IPv4/IPv6 payload packet header is used to forward the packet, and the TTL of the payload packet is decremented. Care should be taken when forwarding such a packet. If the destination address of the payload packet is the encapsulator of the packet (that is the other end of the tunnel), looping can occur. In such a case, the packet must be discarded.

IPv6 forwarding over GRE tunnels

IPv6 forwarding over GRE is accomplished by IPv6 forwarding over IPv4 GRE tunnels. The functionality is similar to the IPv4 forwarding over GRE tunnels (as described above). In the case of IPv6, the FIM (Forward Information Base) module needs to confirm if the forwarding chain is correctly setup both in slowpath and hardware to send the IPv6 packet as a payload of IPv4 GRE encapsulated packet.

Default Router Preferences and More-Specific Routes (RFC 4191)

In some network topologies where the host has multiple routers on its Default Router List, the choice of router for an off-link destination is important. In some situations, one router may provide much better performance than another for a destination. In other situations, choosing the wrong router may result in a failure to communicate.

RFC 4191 describes concepts to use Router Advertisement (RA) messages for communicating default router preferences and more-specific routes from routers to hosts in the IPv6 neighbor discovery (ND) process. This improves the ability of hosts to pick an appropriate router, especially when the host is multi-homed and the routers are on different links. The preference values and specific routes advertised to hosts require administrative configuration; they are not automatically derived from routing tables.

RA messages are used instead of some other protocol like RIP (RFC 2080) because Router Advertisements are an existing standard, stable protocol for router-to-host communication. Piggybacking this information on existing message traffic from routers to hosts reduces network overhead.

Default Router Preferences for Traffic Engineering

Hosts discover and select default devices by listening to Router Advertisements (RAs). Typical default device selection mechanisms are sub-optimal in certain cases, such as when traffic engineering is needed. For example, two devices on a link may provide equivalent but not equal-cost routing, and policy may dictate that one of the devices is preferred. Some examples are as follows:

- Accidentally deploying a new device--Deploying a new device before it has been fully configured could lead to hosts adopting the new device as a default device and traffic disappearing. Network managers may want to indicate that some devices are more preferred than others.

- Multi-homed situations--Multi-homed situations may become more common, because of multiple physical links and because of the use of tunneling for IPv6 transport. Some of the devices may not provide full default routing because they route only to the 6-to-4 prefix or they route only to a corporate intranet. These situations cannot be resolved with redirects, which operate only over a single link.

The default router preference (DRP) feature provides a basic preference metric (low, medium, or high) for default devices. The DRP of a default device is signaled in unused bits in RA messages. This extension is backward compatible, both for devices (setting the DRP bits) and hosts (interpreting the DRP bits). These bits are ignored by hosts that do not implement the DRP extension. Similarly, the values sent by devices that do not implement the DRP extension will be interpreted by hosts that do implement it as indicating a "medium" preference. DRPs need to be configured manually.

Configuring Default Router Preferences

In the IPv6 neighbor discovery process using Router Advertisement (RA) messages, a host has a choice to select the default router based on the router-preference option. For more details of related concepts, see RFC 4191.

Configuration Highlights

- This configuration is not allowed for management interfaces.

- You can set Default Router Preference (DRP) for a router on a specific interface. The router-preference value can be high, medium or low.

- You can set Router Lifetime, which is the length of time in units of seconds for which this router acts as a default router. When ceasing to be an advertising interface, the router should set Router Lifetime to zero in RA message and Route Information options. If the Router Lifetime is set to zero, then the host will not treat the router as the default router. In addition, the DRP configuration will be ignored by the host.

Configuration

The following is a sample configuration of DRP extension to RAs and Router Lifetime:

```
router# configure
router(config)# interface GigabitEthernet 0/2/0/0

router(config-if)# ipv6 nd ra-lifetime 9000
router(config-if)# ipv6 nd router-preference high
```

Use the **no ipv6 nd router-preference** command to remove DRP for a device on a specific interface.

Examples

The following is an example of the configuration:

```
router# show running-config

Wed Jun 29 17:56:26.583 IST
Building configuration...
interface GigabitEthernet0/2/0/0
ipv6 nd ra-lifetime 9000
ipv6 nd router-preference high
!
```

The following is an example of a similar configuration for interfaces part of a Virtual Router Redundancy Protocol (VRRP) group:

```
router# router vrrp
/* Interface part of VRRP group */
interface GigabitEthernet0/2/0/0
address-family ipv6
vrrp 13
priority 110
preempt delay 120
address global 2001:4898:e8:1011:ff::1
address global 2001:4898:e8:1013:ff::1
address global 2001:4898:e8:1014:ff::1
address linklocal autoconfig

/* This interface, which is part of VRRP, has RA configurations */
interface GigabitEthernet0/2/0/0
/* Default Router Lifetime */
ipv6 nd ra-lifetime 9000
/* Default Router Preference */
ipv6 nd router-preference high
!
```

Verification

Use the following command to verify the DRP configuration:

```
router# show ipv6 nd idb interface gigabitEthernet 0/2/0/0 detail location 0/2/CPU0
Mon Jul  4 14:28:53.422 IST

 ifname: Gi0/2/0/0, ifh: 0x01000300, iftype: 15, VI-type: 0, Pseudo IDB: FALSE
 vrf-id: 0x60000000, table-id: 0xe0800000
 Mac Addr: 02d1.1e2b.0baf, size: 6, VLan tag set: FALSE

 IPV6 Interface: Enabled, IPV6: Enabled,  MPLS: Disabled
 Link local address: fe80::d1:1eff:fe2b:baf, Global Addr count: 1
 Global Addresses:1::1(0x2),
 Default Prefix Address: ::, Prefix Addr Count: 3,
 Prefix addresses: 1::(0x401), 2002:4898:e8:1011::(0x4), 2002:4899:e8:1011::(0x4)

RA Specific Route Count: 1,
RA Specific Route : Address 3:: Prefix Length 116 Lifetime 1112 Preference Low

 DAD Attempts: 1, DAD pending 1,
 DAD addresses: fe80::d1:1eff:fe2b:baf

 RA flag: 0x1, Unicast RA send: FALSE, Initial RA count: 3, RA pkts sent count: 0
 Initial RA interval: 16000 msec, Time of Last RA sent: 0, Managed RA timer: 157535048
 RA Managed flag 0x0, RA Other flag 0x0, RA Hop limit 0x0
```

```
 RA Suppress MTU: 0x0, RA Lifetime: 9000 sec
 RA interval min: 160000, max 240000 msec
 RA Router Preference: High
```

Configuring More-Specific Routes

In the IPv6 neighbor discovery process using Router Advertisement (RA) messages, a host is given a route preference based on the RA specific routes option. For more details of related concepts, see RFC 4191.

Configuration Highlights

- This configuration is not allowed for management interfaces.

- A specific route prefix can be set, which is a variable-length field containing an IP address or a prefix of an IP address.

- When the Lifetime is set to zero, the host will no longer use the router for specific route portion of the Route Information options.

- Setting the Preference (using the **preference** keyword) is optional. If no preference value is specified, then the default value for preference (medium) is used.

- Consider a specific route that is already configured, such as **3::3/112** in the following example:

```
reouter#show running-config interface gigabitEthernet 0/2/0/0
interface GigabitEthernet0/2/0/0
ipv6 nd ra specific route 3::3/112 Lifetime 4294967295 preference medium
ipv6 address 1::1/64
!
```

  If any of the three parameters in the configuration (Prefix Length, Lifetime, Preference) is/are changed, the existing specific route (3::3/112) is updated. A new specific route is not created.

- Routers must not include two route information options with the same specific route prefix and prefix length in the same RA.

Configuration

The following is a sample configuration of specific routes:

```
router# configure
router(config)# interface GigabitEthernet 0/2/0/0
router(config-if)# ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
router(config-if)# ipv6 address 1::1/64
```

Use the **no ipv6 nd ra specific route** *prefix* command to delete a single specific route under an interface. Use the **no ipv6 nd ra specific route** command to delete all specific routes under an interface.

Examples

The following is an example of the configuration:

```
router# show running-config

Wed Jun 29 17:56:26.583 IST
Building configuration...
interface GigabitEthernet0/2/0/0
```

```
ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
ipv6 address 1::1/64
!
```

The following is an example of a similar configuration for interfaces part of a Virtual Router Redundancy Protocol (VRRP) group:

```
router# router vrrp
/* Interface part of VRRP group */
interface GigabitEthernet0/2/0/0
address-family ipv6
vrrp 13
priority 110
preempt delay 120
address global 2001:4898:e8:1011:ff::1
address global 2001:4898:e8:1013:ff::1
address global 2001:4898:e8:1014:ff::1
address linklocal autoconfig

/* This interface, which is part of VRRP, has RA configurations */
interface GigabitEthernet0/2/0/0
/* Specific Route configuration */
ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
ipv6 address 1::1/64
!
```

Verification

Use the following command to verify the specific routes configuration:

```
router# show ipv6 nd idb interface gigabitEthernet 0/2/0/0 detail location 0/2/CPU0
Mon Jul  4 14:28:53.422 IST

 ifname: Gi0/2/0/0, ifh: 0x01000300, iftype: 15, VI-type: 0, Pseudo IDB: FALSE
 vrf-id: 0x60000000, table-id: 0xe0800000
 Mac Addr: 02d1.1e2b.0baf, size: 6, VLan tag set: FALSE

 IPV6 Interface: Enabled, IPV6: Enabled,  MPLS: Disabled
 Link local address: fe80::d1:1eff:fe2b:baf, Global Addr count: 1
 Global Addresses:1::1(0x2),
 Default Prefix Address: ::, Prefix Addr Count: 3,
 Prefix addresses: 1::(0x401), 2002:4898:e8:1011::(0x4), 2002:4899:e8:1011::(0x4)

 RA Specific Route Count: 1,
 RA Specific Route : Address 3:: Prefix Length 116 Lifetime 1112 Preference Low

 DAD Attempts: 1, DAD pending 1,
 DAD addresses: fe80::d1:1eff:fe2b:baf

 RA flag: 0x1, Unicast RA send: FALSE, Initial RA count: 3, RA pkts sent count: 0
 Initial RA interval: 16000 msec, Time of Last RA sent: 0, Managed RA timer: 157535048
 RA Managed flag 0x0, RA Other flag 0x0, RA Hop limit 0x0
 RA Suppress MTU: 0x0, RA Lifetime: 9000 sec
 RA interval min: 160000, max 240000 msec
 RA Router Preference: High
```

Troubleshooting Default Router Preferences and More-Specific Routes

Default Router Preference (DRP) and specific routes are sent as part of RA messages.

The interface for which DRP and specific routes are configured, should be IPv6 enabled and the IPv6 protocol should be up. Verify using the following command:

```
router# show ipv6 interface gigabitEthernet 0/2/0/0
Wed Jun 29 18:52:31.936 IST
GigabitEthernet0/2/0/0 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
IPv6 is enabled, link-local address is fe80::d1:1eff:fe2b:baf
  Global unicast address(es):
    1::1, subnet is 1::/64
  Joined group address(es): ff02::1:ff00:1 ff02::1:ff2b:baf ff02::2
      ff02::1
```

If the interface has IPv6 protocol up and the RAs are not being sent, then verify whether RA is suppressed:

```
router#show running-config interface gigabitEthernet 0/2/0/0
interface GigabitEthernet0/2/0/0
ipv6 nd suppress-ra <<< configuration of RA suppression under interface
 ipv6 nd router-preference high
 ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
 ipv6 address 1::1/64
!
```

If the interface has IPv6 protocol up and the RAs are not being sent, then verify whether RA is enabled in Interface Descriptor Block (IDB). The RA flag is 1.

```
router#show ipv6 nd idb  interface gigabitEthernet 0/2/0/0 detail location 0/2/CPU0
Wed Jun 29 21:50:27.255 IST

RA Specific Route Count: 1,
RA Specific Route : Address 3:: Prefix Length 116 Lifetime 1112 Preference Low <<< specific
 route for interface

RA DNS Servers Addr Count: 0,

RA DNS Search List Count: 0,

DAD Attempts: 1, DAD pending 0,

RA flag: 0x1, Unicast RA send: FALSE, Initial RA count: 3, RA pkts sent count: 3039 <<< RA
 flag should be 1
```

Run the IPv6 ND traces to debug any particular issue related to a specific route:

```
router#show ipv6 nd trace location 0/2/CPU0
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702 Sending RA to ff02::1 on GigabitEthernet0/2/0/0
 (0x1000300)
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702  hoplimit 64 lifetime 9000 reachable 0 retrans
 0
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702   1::/64 Onlink Auto
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702    valid 2592000 pref 604800
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702   2002:4898:e8:1011::/64 Onlink Auto
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702    valid 1111 pref 222
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702   2002:4899:e8:1011::/64 Onlink Auto
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702    valid 1111 pref 222
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702 ra specific route address  3:: lifetime 1112
 preference Low
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702 nd_send_ra: sending RA paksize=192, plen=152
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702 nd_pak_send: size=192, ifh
GigabitEthernet0/2/0/0 (0x1000300) , priority=2 to ipv6-io
Jun 29 21:53:41.673 nd/fevent 0/2/CPU0 t26702 nd_pak_send: sending pak=0x60c07d8b with NO
FVS set, size=192, ifh GigabitEthernet0/2/0/0 (0x1000300)  to ipv6-io
```

IPv6 Router Advertisement Options for DNS Configuration (RFC 6106)

DNS configuration based on IPv6 Router Advertisement (RA) options is useful in networks where:

1. An IPv6 host's address is autoconfigured through IPv6 stateless address autoconfiguration.

2. There is either no DHCP IPv6 infrastructure at all or some hosts do not have a DHCPv6 client.

In these scenarios, it is not feasible to manually configure hosts each time they connect to a different network. While a one-time static configuration is possible, it is generally not desirable on general-purpose hosts such as laptops and mobile phones.

Neighbor Discovery (ND) for IPv6 and IPv6 stateless address autoconfiguration provide ways to configure either fixed or mobile nodes with one or more IPv6 addresses, default routers, and some other parameters [RFC4861][RFC4862]. Most Internet services are identified by using a DNS name. The following two RA options defined in RFC 6106 provide the DNS information needed for an IPv6 host to reach Internet services:

- **Recursive DNS Server (RDNSS) Option** – RDNSS provides a recursive DNS resolution service for translating domain names into IP addresses as defined in [RFC1034] and [RFC1035]. The RDNSS option contains one or more IPv6 addresses of recursive DNS servers. All addresses share the same Lifetime value, which is the maximum time in seconds over which this RDNSS address may be used for name resolution. If different Lifetime values are required, multiple RDNSS options can be used. There can be up to 5 DNS servers.

- **DNS Search List (DNSSL) Option** – DNSSL is a list of DNS suffix domain names used by IPv6 hosts when they perform DNS query searches for short, unqualified domain names. The DNSSL option contains one or more domain names. All domain names share the same Lifetime value, which is the maximum time in seconds over which this DNSSL may be used. If different Lifetime values are required, multiple DNSSL options can be used. There can be up to 50 DNSSLs.

Note If DNS information is available from multiple RAs and/or from DHCP, the host must maintain an ordered list of this DNS information.

Configuring DNS Server Using IPv6 Router Advertisement Options

RFC 6106 specifies IPv6 Router Advertisement (RA) options to allow IPv6 routers to advertise a list of recursive DNS server (RDNSS) addresses used for the DNS name resolution in IPv6 hosts.

The DNS lifetime range should be between maximum RA interval and twice the maximum RA interval:

```
(max ra interval) <= dns lifetime <= (2*(max ra interval))
```

The maximum RA interval can have a value between 4 and 1800 seconds (default is 240 seconds). Therefore, the following configuration will result in a lifetime out of range:

```
router(config-if)#ipv6 nd ra dns server 4::4 3600 <<< Lifetime configured out of range for
 the interface having default maximum RA interval
```

Configuration

The following is a sample configuration of recursive DNS server:

```
router# configure
router(config)# interface GigabitEthernet 0/2/0/0

router(config-if)# ipv6 nd prefix 2002:4898:e8:1011::/64 1111 222
router(config-if)# ipv6 nd prefix 2002:4899:e8:1011::/64 1111 222
router(config-if)# ipv6 nd ra-lifetime 9000


router(config-if)# ipv6 nd ra dns server 4::4 infinite-lifetime
router(config-if)# ipv6 nd router-preference high
router(config-if)# ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
```

Use the **no ipv6 nd ra dns server** *ipv6 address* command to delete a single DNS server under an interface. Use the **no ipv6 nd ra dns server** command to delete all DNS servers under an interface.

Example

The following is an example of the DNS server configuration for a specific interface:

```
router# show running-config interface gigabitEthernet 0/2/0/0

interface GigabitEthernet0/2/0/0
ipv6 nd prefix 2002:4898:e8:1011::/64 1111 222
 ipv6 nd prefix 2002:4899:e8:1011::/64 1111 222
 ipv6 nd ra-lifetime 9000
/* RDNSS configuration */
ipv6 nd ra dns server 4::4 infinite-lifetime
ipv6 nd router-preference high
ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
!
```

Verification

Use the following command to verify the DNS server configuration based on IPv6 RA options:

```
router# show ipv6 nd idb interface gigabitEthernet 0/2/0/0 detail location 0/2/CPU0
Mon Jul  4 14:28:53.422 IST

 ifname: Gi0/2/0/0, ifh: 0x01000300, iftype: 15, VI-type: 0, Pseudo IDB: FALSE
 vrf-id: 0x60000000, table-id: 0xe0800000
 Mac Addr: 02d1.1e2b.0baf, size: 6, VLan tag set: FALSE

 Media Name: ether, Media Encap: 0x1 (ARPA)
 Mac Length: 6, Media Header Len: 14, Media Proto: 0xdd86
 Current  Encap: 0x1 (ARPA), Mcast  Encap : 0x1 (ARPA)

 IPV6 Interface: Enabled, IPV6: Enabled,  MPLS: Disabled
 Link local address: fe80::d1:1eff:fe2b:baf, Global Addr count: 1
 Global Addresses:1::1(0x2),
 Default Prefix Address: ::, Prefix Addr Count: 3,
 Prefix addresses: 1::(0x401), 2002:4898:e8:1011::(0x4), 2002:4899:e8:1011::(0x4)

 RA Specific Route Count: 1,
 RA Specific Route : Address 3:: Prefix Length 116 Lifetime 1112 Preference Low

 RA DNS Servers Addr Count: 3,
 RA DNS Server addresse: Address 5::6 Lifetime 240
```

```
 RA DNS Server addresse: Address 5::5 Lifetime 240
 RA DNS Server addresse: Address 4::4 Lifetime 4294967295
```

Configuring DNS Search List Using IPv6 Router Advertisement Options

RFC 6106 specifies IPv6 Router Advertisement (RA) options to allow IPv6 routers to advertise a DNS Search List (DNSSL) to IPv6 hosts for an enhanced DNS configuration.

The DNS lifetime range should be between maximum RA interval and twice the maximum RA interval:

```
(max ra interval) <= dns lifetime <= (2*(max ra interval))
```

The maximum RA interval can have a value between 4 and 1800 seconds (default is 240 seconds). Therefore, the following configuration will result in a lifetime out of range:

```
router(config-if)#ipv6 nd ra dns search list sss.com 3600 <<< Lifetime configured out of
range for the interface having default maximum RA interval
```

Configuration

The following is a sample configuration of a DNS search list:

```
router# configure
router(config)# interface GigabitEthernet 0/2/0/0

router(config-if)# ipv6 nd prefix 2002:4898:e8:1011::/64 1111 222
router(config-if)# ipv6 nd prefix 2002:4899:e8:1011::/64 1111 222
router(config-if)# ipv6 nd ra-lifetime 9000

router(config-if)# ipv6 nd ra dns search list aaa.cc.com infinite-lifetime
router(config-if)# ipv6 address 1::1/64
```

✎

Note   The domain name configuration should follow RFC 1035. If not, the configuration will be rejected. For example, the following domain name configuration will result in an error:

```
router(config-if)#ipv6 nd ra dns search list .aaa.cc.com infinite-lifetime
```

Use the **no ipv6 nd ra dns search list** *name* command to delete a single DNS search list under an interface. Use the **no ipv6 nd ra dns search list** command to delete all DNS search lists under an interface.

Example

The following is an example of the configuration for a specific interface:

```
router# show running-config interface gigabitEthernet 0/2/0/0

interface GigabitEthernet0/2/0/0
ipv6 nd prefix 2002:4898:e8:1011::/64 1111 222
 ipv6 nd prefix 2002:4899:e8:1011::/64 1111 222
 ipv6 nd ra-lifetime 9000
ipv6 nd router-preference high
ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low

/* DNSSL configuration */
ipv6 nd ra dns search list aaa.cc.com infinite-lifetime
```

```
ipv6 address 1::1/64
!
```

Verification

Use the following command to verify the DNS search list configuration based on IPv6 RA options:

```
router# show ipv6 nd idb interface gigabitEthernet 0/2/0/0 detail location 0/2/CPU0
Mon Jul  4 14:28:53.422 IST

 ifname: Gi0/2/0/0, ifh: 0x01000300, iftype: 15, VI-type: 0, Pseudo IDB: FALSE
 vrf-id: 0x60000000, table-id: 0xe0800000
 Mac Addr: 02d1.1e2b.0baf, size: 6, VLan tag set: FALSE

 Media Name: ether, Media Encap: 0x1 (ARPA)
 Mac Length: 6, Media Header Len: 14, Media Proto: 0xdd86
 Current  Encap: 0x1 (ARPA), Mcast  Encap : 0x1 (ARPA)

 IPV6 Interface: Enabled, IPV6: Enabled,  MPLS: Disabled
 Link local address: fe80::d1:1eff:fe2b:baf, Global Addr count: 1
 Global Addresses:1::1(0x2),
 Default Prefix Address: ::, Prefix Addr Count: 3,
 Prefix addresses: 1::(0x401), 2002:4898:e8:1011::(0x4), 2002:4899:e8:1011::(0x4)

 RA Specific Route Count: 1,
 RA Specific Route : Address 3:: Prefix Length 116 Lifetime 1112 Preference Low

 RA DNS Search List Count: 3,
 RA DNS Search List : Name aaa.bbb.ccc.com Lifetime 240
 RA DNS Search List : Name aaa.bbb.com Lifetime 240
 RA DNS Search List : Name aaa.cc.com Lifetime 4294967295
```

Troubleshooting DNS Servers and DNS Search Lists

Recursive DNS servers and DNS search lists are sent as part of RA messages.

The interface for which the DNS servers and DNS search lists are configured, should be IPv6 enabled and the IPv6 protocol should be up. Verify using the following command:

```
router# show ipv6 interface gigabitEthernet 0/2/0/0
Wed Jun 29 18:52:31.936 IST
GigabitEthernet0/2/0/0 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
IPv6 is enabled, link-local address is fe80::d1:1eff:fe2b:baf
  Global unicast address(es):
    1::1, subnet is 1::/64
  Joined group address(es): ff02::1:ff00:1 ff02::1:ff2b:baf ff02::2
      ff02::1
```

If the interface has IPv6 protocol up and the RAs are not being sent, then verify whether RA is suppressed:

```
router#show running-config interface gigabitEthernet 0/2/0/0
interface GigabitEthernet0/2/0/0
ipv6 nd suppress-ra <<< configuration of RA suppression under interface
 ipv6 nd router-preference high
 ipv6 nd ra specific route 3::3/116 Lifetime 1112 preference low
 ipv6 address 1::1/64
!
```

If the interface has IPv6 protocol up and the RAs are not being sent, then verify whether RA is enabled in Interface Descriptor Block (IDB). The RA flag is 1.

```
router#show ipv6 nd idb  interface gigabitEthernet 0/2/0/0 detail location 0/2/CPU0 Thu Jun
 30 20:02:20.211 IST

ifname: Gi0/2/0/0, ifh: 0x01000300, iftype: 15, VI-type: 0, Pseudo IDB: FALSE
vrf-id: 0x60000000, table-id: 0xe0800000
Mac Addr: 02d1.1e2b.0baf, size: 6, VLan tag set: FALSE

RA Specific Route Count: 1,
RA Specific Route : Address 3:: Prefix Length 116 Lifetime 1112 Preference Low

RA DNS Servers Addr Count: 1,
RA DNS Server addresse: Address 4::4 Lifetime 4294967295 <<< RDNSS for interface

RA DNS Search List Count: 1,
RA DNS Search List : Name aaa.cc.com Lifetime 4294967295 <<< DNSSL for interface

 RA flag: 0x1, Unicast RA send: FALSE, Initial RA count: 3, RA pkts sent count: 442 <<< RA
 flag should be 1
```

Run the IPv6 ND traces to debug any particular issue related to a DNS servers and DNS search lists:

```
router#show ipv6 nd trace location 0/2/CPU0
Jun 30 20:07:03.508 nd/fevent 0/2/CPU0 t26702 Sending RA to ff02::1 on GigabitEthernet0/2/0/0
 (0x1000300)
Jun 30 20:07:03.508 nd/fevent 0/2/CPU0 t26702   hoplimit 64 lifetime 9000 reachable 0 retrans
 0
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702   1::/64 Onlink Auto
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702    valid 2592000 pref 604800
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702   2002:4898:e8:1011::/64 Onlink Auto
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702    valid 1111 pref 222
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702   2002:4899:e8:1011::/64 Onlink Auto
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702    valid 1111 pref 222
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra specific route address  3:: lifetime 1112
 preference Low
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra dns server address  5::6 lifetime 240 first
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra dns server address  5::5 lifetime 240 part
 of same ra dns server option
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra dns server address  4::4 lifetime 4294967295
 first
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra dns search list name aaa.bbb.ccc.com
lifetime 240 first
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra dns search list name aaa.bbb.com lifetime
 240 part of same ra dns search list option
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 ra dns search list name aaa.cc.com lifetime
4294967295 first
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 nd_send_ra: sending RA paksize=320, plen=280
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 nd_pak_send: size=320, ifh
GigabitEthernet0/2/0/0 (0x1000300) ,  priority=2 to ipv6-io
Jun 30 20:07:03.509 nd/fevent 0/2/CPU0 t26702 nd_pak_send: sending pak=0x60c07d8b with NO
FVS set, size=320, ifh GigabitEthernet0/2/0/0 (0x1000300)  to ipv6-io
```

TCP MSS Adjustment

The TCP Maximum Segment Size (MSS) Adjustment feature allows the configuration of Maximum Segment Size (MSS) on transient packets that traverse a router. TCP MSS Adjustment is used on a GRE tunnel interface or VLAN sub-interface on a physical Ethernet interface on ASR 9000 Enhanced Ethernet Line Card to enable the transit traffic of TCP flows to be a Maximum Segment Size (MSS) below the GRE tunnel interface or

VLAN sub-interface MTU so that traffic fragmentation is prevented when a session is established. It needs to be configurable, for a single tunnel interface or VLAN sub-interface, to a specific value.

This feature is supported for transit IPv4 and IPv6 packets only. It applies to both ingress and egress traffic on interfaces where the TCP MSS adjustment is configured. In this release, a configuration of **ipv4 tcp-mss-adjust enable** or **ipv6 tcp-mss-adjust enable** or both commands on an interface will have the same effect. It applies to all TCP SYNC packets encapsulated in an IPv4 or IPv6 frame, coming in and going out of the interface.

Configuring TCP MSS for IPv4 packets

This task describes how to enable the transit traffic of TCP flows to be a Maximum Segment Size (MSS) below the GRE tunnel interface or VLAN sub-interface MTU so that traffic fragmentation is prevented when a session is established for IPv4 packets.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 tcp-mss-adjust enable**
4. **commit**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure**	

Step 2	**interface** *type interface-path-id*
	
	 Example:
	
	 `RP/0/RSP0/CPU0:router(config)# interface`
	 `gigabitethernet 0/2/0/0.100`
		Enters interface configuration mode and configures an interface.

Step 3	**ipv4 tcp-mss-adjust enable**
	
	 Example:
	
	 `RP/0/RSP0/CPU0:router(config-if)# ipv4`
	 `tcp-mss-adjust enable`
		Enable the transit traffic of TCP flows to be a Maximum Segment Size (MSS) below the GRE tunnel interface or VLAN sub-interface MTU so that traffic fragmentation is prevented when a session is established for IPv4 packets.

Step 4	**commit**	

Configuring TCP MSS for IPv6 packets

This task describes how to enable the transit traffic of TCP flows to be a Maximum Segment Size (MSS) below the GRE tunnel interface or VLAN sub-interface MTU so that traffic fragmentation is prevented when a session is established for IPv6 packets.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv6 tcp-mss-adjust enable**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/2/0/0.100	Enters interface configuration mode and configures an interface.
Step 3	**ipv6 tcp-mss-adjust enable** **Example:** RP/0/RSP0/CPU0:router(config-if)# ipv6 tcp-mss-adjust enable	Enables the transit traffic of TCP flows to be a Maximum Segment Size (MSS) below the GRE tunnel interface or VLAN sub-interface MTU so that traffic fragmentation is prevented when a session is established for IPv6 packets.
Step 4	**commit**	

Configuration Examples for Implementing Network Stack IPv4 and IPv6

This section provides the following configuration examples:

Assigning an Unnumbered Interface: Example

In the following example, the second interface (GigabitEthernet 0/1/0/1) is given the address of loopback interface 0. The loopback interface is unnumbered.

```
interface loopback 0
 ipv4 address 192.168.0.5 255.255.255.0
interface gigabitethernet 0/1/0/1
 ipv4 unnumbered loopback 0
```

Configuring VRF mode big

Perform this task to configure big mode for a VRF.

SUMMARY STEPS

1. **configure**
2. **vrf** *vrf-name*
3. **mode big**
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**vrf** *vrf-name* **Example:** RP/0/RSP0/CPU0:router(config)# vrf v1 RP/0/RSP0/CPU0:router(config-vrf)#	Enters the vrf configuration mode.
Step 3	**mode big** **Example:** RP/0/RSP0/CPU0:router(config-vrf)# mode big RP/0/RSP0/CPU0:router(config-vrf)#	Enters the big mode for the corresponding VRF.
Step 4	**commit**	

Additional References

The following sections provide references related to implementing Network Stack IPv4 and IPv6.

Related Documents

Related Topic	Document Title
Address resolution configuration tasks	*Configuring ARP* module in this publication.
Mapping host names to IP addresses	*Host Services and Applications Commands* module in the *IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*
Network stack IPv4 and IPv6 commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Network Stack IPv4 and IPv6 Commands* section in the *IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport